Pattern Recognition

Bertrand Thirion and John Ashburner

General setting

We have a training dataset of n observations, each consisting of an input $\mathbf{x}_i$ and a target y_i.
Each input, $\mathbf{x}_i$, consists of a vector of p features.

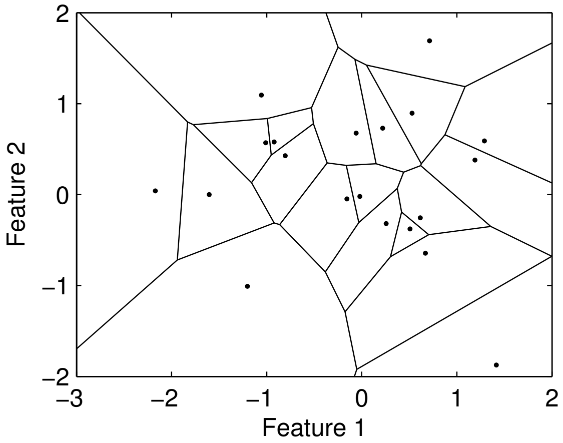
$$\mathcal{D} = \{(\mathbf{x}_i, y_i) | i = 1, .., n\}$$

The aim is to predict the target for a new input $\mathbf{x}_*$.

Curse of dimensionality

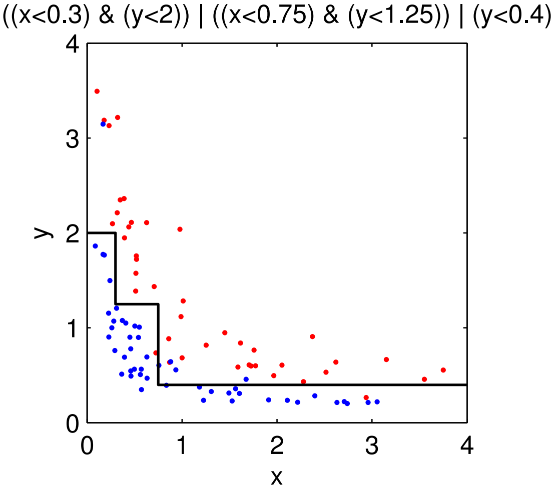
Large p, small n.

Nearest-neighbour classification



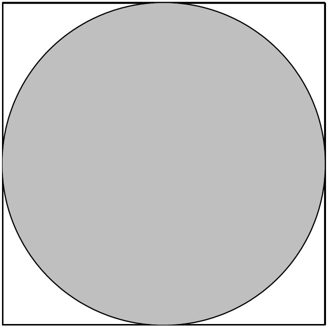
- Not nice smooth separations.
- Lots of sharp corners.
- May be improved with *K-nearest neighbours*.

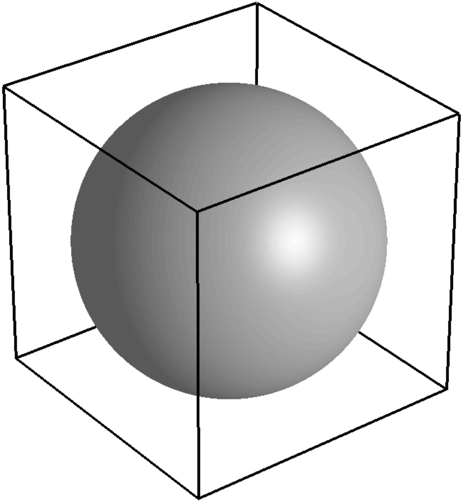
Rule-based approaches



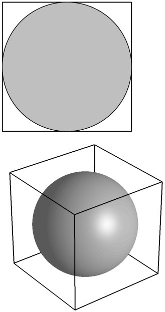
- Not nice smooth separations.
- Lots of sharp corners.

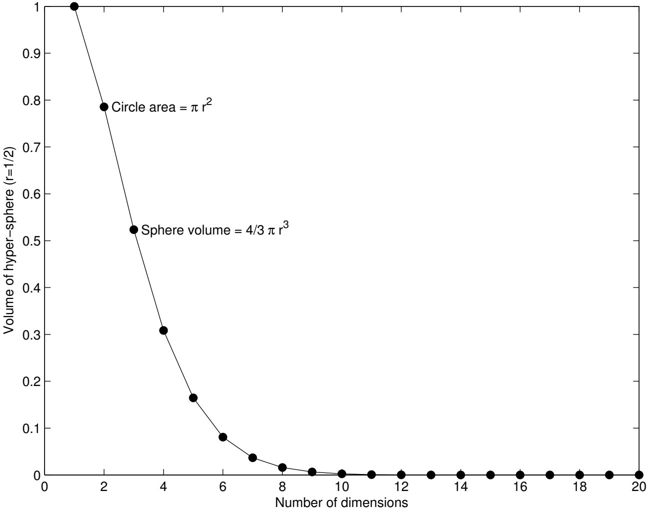
Corners matter in high-dimensions





Corners matter in high-dimensions





Occam's razor

"Everything should be kept as simple as possible, but no simpler."

— Einstein (allegedly)

- Complex models (with many estimated parameters) usually explain training data better than simpler models.
- Simpler models often generalise better to new data than nore complex models.

Need to find the model with the optimal bias/variance tradeoff.

Bayesian model selection

Real Bayesians don't cross-validate (except when they need to).

$$P(M|\mathcal{D}) = \frac{p(\mathcal{D}|M)P(M)}{P(\mathcal{D})}$$

The *Bayes factor* allows the plausibility of two models (M_1 and M_2) to be compared:

$$K = \frac{p(\mathcal{D}|M_1)}{p(\mathcal{D}|M_2)} = \frac{\int_{\theta_{M_1}} p(\mathcal{D}|\theta_{M_1}, M_1)p(\theta_{M_1}|M_1)d\theta_{M_1}}{\int_{\theta_{M_2}} p(\mathcal{D}|\theta_{M_2}, M_2)p(\theta_{M_2}|M_2)d\theta_{M_2}}$$

This is usually too costly in practice, so approximations are used.

Model selection

Some approximations/alternatives to the Bayesian approach:

- **Laplace approximations**: find the MAP/ML solution and use a Gaussian approximation to the parameter uncertainty.
- **Minimum Message Length** (MML): an information theoretic approach.
- **Minimum Description Length** (MDL): an information theoretic approach based on how well the model compresses the data.
- **Akaike Information Criterion** (AIC): $-2 \log p(\mathcal{D}|\theta) + 2k$, where k is the number of estimated parameters.
- **Bayesian Information Criterion** (BIC): $-2 \log p(\mathcal{D}|\theta) + k \log q$, where q is the number of observations.

Log predictive probability

Some data are more easily classified than others.
Probabilistic classifiers provide a level of confidence for each prediction.

$$p(y_*|\mathbf{x}_*, \mathbf{y}, \mathbf{X}, \theta)$$

Quality of predictions can be assessed using the **test log predictive probability**:

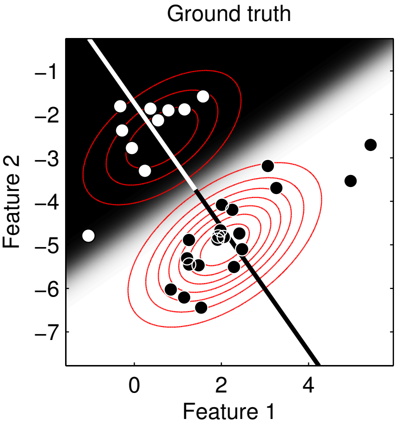
$$\frac{1}{m}\sum_{i=1}^{m}\log_2 p(y_{*i}=t_i|\mathbf{x}_{*i}, \mathbf{y}, \mathbf{X}, \theta)$$

After subtracting the baseline measure, this shows the average bits of information given by the model.

Rasmussen & Williams. "Gaussian Processes for Machine Learning", MIT Press (2006).
`http://www.gaussianprocess.org/gpml/`

Generative models for classification

$$P(y=k|\mathbf{x}) = \frac{P(y=k)p(\mathbf{x}|y=k)}{\sum_j P(y=j)p(\mathbf{x}|y=j)}$$

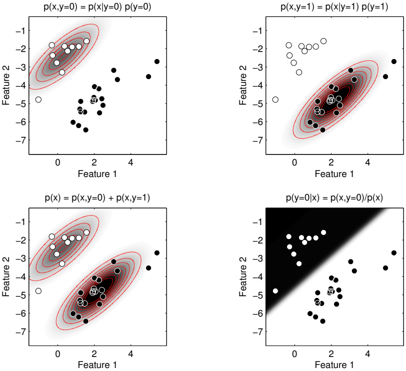


Linear discriminant analysis

$$P(y=k|\mathbf{x}) = \frac{P(y=k)p(\mathbf{x}|y=k)}{\sum_j P(y=j)p(\mathbf{x}|y=j)}$$

Assumes:

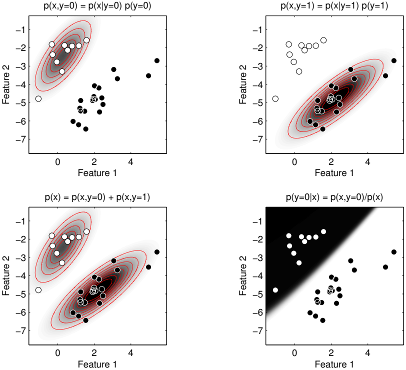
$$P(\mathbf{x}|y=k) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$$



Model has $2p + p(p-1)$ parameters to estimate (two means and a single covariance).

Number of observations is pn (size of inputs).

Quadratic discriminant analysis



$$P(y=k|\mathbf{x}) = \frac{P(y=k)p(\mathbf{x}|y=k)}{\sum_j P(y=j)p(\mathbf{x}|y=j)}$$

Assumes different covariances:

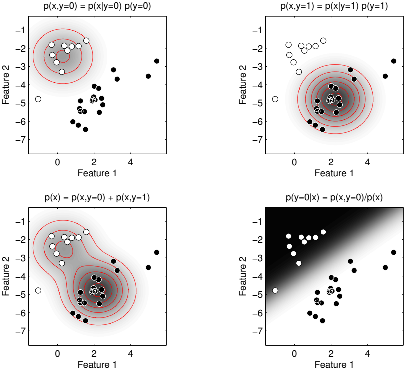
$$P(\mathbf{x}|y=k) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Model has $2p + 2p(p-1)$ parameters to estimate (two means and two covariances).

Number of observations is pn.

Naive Bayes

$$P(y=k|\mathbf{x}) = \frac{P(y=k)p(\mathbf{x}|y=k)}{\sum_j P(y=j)p(\mathbf{x}|y=j)}$$



Assumes that features are independent:

$$p(\mathbf{x}|y=k) = \prod_i p(x_i|y=k)$$

Model has variable number of parameters to estimate, but the above example has $3p$.

Number of observations is pn.

Linear regression: maximum likelihood

A simple way to do regression is by:

$$f(\mathbf{x}_*) = \mathbf{a}^T\mathbf{x}_*$$

Assuming Gaussian noise on **y**, the ML estimate of **a** is by:

$$\hat{\mathbf{a}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

where

$$\mathbf{X} = \begin{pmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_n\end{pmatrix}^T, \text{ and } \mathbf{y} = \begin{pmatrix}y_1 & y_2 & \dots y_n\end{pmatrix}^T$$

Model has p parameters to estimate.
Number of observations is n (number of targets).
Usualy needs dimensionality reduction, with (eg) SVD.

Linear regression: maximum posterior

We may have prior knowledge about various distributions:

$$p(y_*|\mathbf{x}_*, \mathbf{a}) = \mathcal{N}(\mathbf{a}^T\mathbf{x}_*, \sigma^2)$$
$$p(\mathbf{a}) = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_0)$$

Therefore,

$$p(\mathbf{a}|\mathbf{y}, \mathbf{X}) = \mathcal{N}(\sigma^{-2}\mathbf{B}^{-1}\mathbf{X}^T\mathbf{y}, \mathbf{B}^{-1}), \text{ where } \mathbf{B} = \sigma^{-2}\mathbf{X}^T\mathbf{X} + \boldsymbol{\Sigma}_0^{-1}$$

Maximum a posteriori (MAP) estimate of **a** is by:

$$\hat{\mathbf{a}} = \sigma^{-2}\mathbf{B}^{-1}\mathbf{X}^T\mathbf{y}, \text{ where } \mathbf{B} = \sigma^{-2}\mathbf{X}^T\mathbf{X} + \boldsymbol{\Sigma}_0^{-1}$$

Linear regression: Bayesian

We may have prior knowledge about various distributions:

$$\begin{aligned} p(y_*|\mathbf{x}_*, \mathbf{a}) =& \mathcal{N}(\mathbf{a}^T\mathbf{x}_*, \sigma^2) \\ p(\mathbf{a}) =& \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_0) \end{aligned}$$

Therefore,

$$p(\mathbf{a}|\mathbf{y}, \mathbf{X}) = \mathcal{N}(\sigma^{-2}\mathbf{B}^{-1}\mathbf{X}^T\mathbf{y}, \mathbf{B}^{-1}), \text{ where } \mathbf{B} = \sigma^{-2}\mathbf{X}^T\mathbf{X} + \mathbf{\Sigma}_0^{-1}$$

Predictions are made by integrating out the uncertainty of the weights, rather than estimating them:

$$\begin{aligned} p(y_*|\mathbf{x}_*, \mathbf{y}, \mathbf{X}) =& \int_{\mathbf{a}} p(y_*|\mathbf{x}_*, \mathbf{a}) p(\mathbf{a}|\mathbf{y}, \mathbf{X}) d\mathbf{a} \\ =& \mathcal{N}(\sigma^{-2}\mathbf{x}_*^T\mathbf{B}^{-1}\mathbf{X}^T\mathbf{y}, \mathbf{x}_*^T\mathbf{B}^{-1}\mathbf{x}_*) \end{aligned}$$

Estimated parameters may be σ^2, and parameters encoding $\mathbf{\Sigma}_0$.

Kernel methods: Woodbury matrix identity

$$\begin{aligned}\mathbf{B}^{-1} &= \left(\sigma^{-2}\mathbf{X}^T\mathbf{X} + \mathbf{\Sigma}_0^{-1}\right)^{-1} \\ &= \mathbf{\Sigma}_0 - \mathbf{\Sigma}_0\mathbf{X}^T(\mathbf{I}\sigma^2 + \mathbf{X}\mathbf{\Sigma}_0\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{\Sigma}_0\end{aligned}$$

Wikipedia contributors, "Woodbury matrix identity," Wikipedia, The Free Encyclopedia, `http://en.wikipedia.org/w/index.php?title=Woodbury_matrix_identity&oldid=638370219` (accessed April 1, 2015).

Dimensions of $\mathbf{X}^T\mathbf{X}$ are $p \times p$.
Dimensions of $\mathbf{X}\mathbf{\Sigma}_0\mathbf{X}^T$ are $n \times n$.

Kernel methods: Gaussian process regression

The predicted distribution is:

$$p(y_* | \mathbf{x}_*, \mathbf{y}, \mathbf{X}) = \mathcal{N}(\mathbf{k}^T \mathbf{C}^{-1} \mathbf{y}, c - \mathbf{k}^T \mathbf{C}^{-1} \mathbf{k})$$

where:

$$\begin{aligned} \mathbf{C} &= \mathbf{X} \boldsymbol{\Sigma}_0 \mathbf{X}^T + \mathbf{I} \sigma^2 \\ \mathbf{k} &= \mathbf{X} \boldsymbol{\Sigma}_0 \mathbf{x}_* \\ c &= \mathbf{x}_*^T \boldsymbol{\Sigma}_0 \mathbf{x}_* + \sigma^2 \end{aligned}$$

Kernel methods: nonlinear methods

Sometimes, we want alternatives to $\mathbf{C} = \mathbf{X}\mathbf{\Sigma}_0\mathbf{X}^T + \mathbf{I}\sigma^2$.
Nonlinearity is achieved by replacing the matrix $\mathbf{K} = \mathbf{X}\mathbf{\Sigma}_0\mathbf{X}^T$ with some function of the data that gives a positive definite matrix encoding similarities.
eg

$$k(\mathbf{x}_i, \mathbf{x}_j) = \theta_1 + \theta_2 \mathbf{x}_i \cdot \mathbf{x}_\mathbf{j} + \theta_3 \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_\mathbf{j}||^2}{2\theta_4^2} \right)$$

Hyper-parameters θ_1 to θ_4 can be optimised in a number of ways.

Kernel methods: nonlinear methods

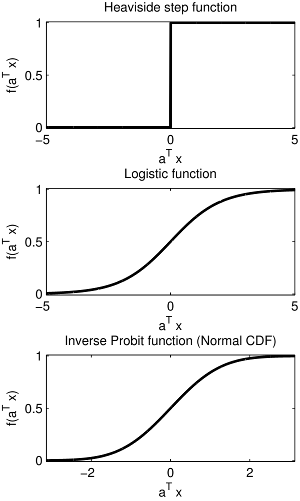
For large p, small n problems, nonlinear methods do not seem to help much.
Nonlinearity also reduces interpretability.

Discriminative models for classification

$$t = \sigma(f(\mathbf{x}_*))$$

where σ is some squashing function, eg:

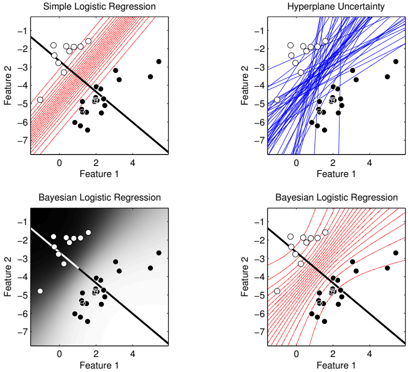
- Heaviside step function.
- Logistic function (inverse of Logit).
- Normal CDF (inverse of Probit).



Probabilistic classification

Integrating over the uncertainty of the separating hyperplane allows probabilistic predictions further from the training data. This is not usually done for methods such as the revelance-vector machine (RVM).

Rasmussen, Carl Edward, and Joaquin Quinonero-Candela. "Healing the relevance vector machine through augmentation." In Proceedings of the 22nd international conference on Machine learning, pp. 689-696. ACM, 2005.



Probabilistic classification

Making probabilistic predictions involves:

1. Computing the distribution of a latent variable corresponding to the test data (cf regression):

$$p(f_*|\mathbf{x}_*, \mathbf{y}, \mathbf{X}) = \int_{\mathbf{f}} p(f_*|\mathbf{x}_*, \mathbf{f})p(\mathbf{f}|\mathbf{y}, \mathbf{X})d\mathbf{f}$$

2. Using this distribution to give a probabilistic prediction:

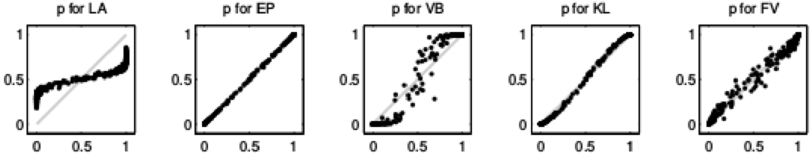
$$P(y_* = 1|\mathbf{x}_*, \mathbf{y}, \mathbf{X}) = \int_{f_*} \sigma(f_*)p(f_*|\mathbf{x}_*, \mathbf{y}, \mathbf{X})df_*$$

Unfortunately, the second integral is analytically intractable, so approximations are needed.

Probabilistic classification

Approximate methods for probabilistic classification include:

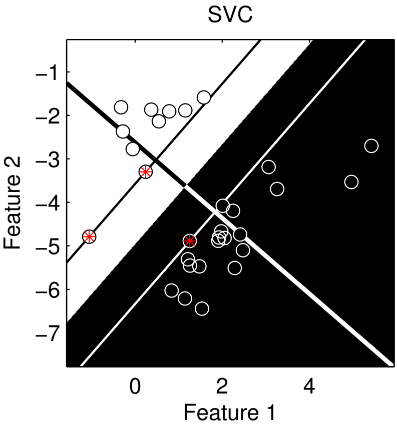
- **The Laplace Approximation** (LA). Fastest, but less accurate.
- **Expectation Propagation** (EP). More accurate than the Laplace approximation, but slightly slower.
- **MCMC** methods. The “gold standard”, but very slow to draw lots of random samples.



Nickisch, Hannes, and Carl Edward Rasmussen. “Approximations for Binary Gaussian Process Classification.” Journal of Machine Learning Research 9 (2008): 2035-2078.

Support vector classification

If you are only interested in binary predictions, support-vector machines are reasonably fast and accurate.



Ensemble learning

Combining predictions from weak learners.

- **Bootstrap aggregating (bagging)**
  - Train several weak classifiers, with different models or randomly drawn subsets of the data.
  - Average their predictions with equal weight.
- **Boosting**
  - A family of approaches, where models are weighted according to their accuracy.
  - AdaBoost is popular, but has problems with target noise.
- **Bayesian model averaging**
  - Really a model selection method.
  - Relatively ineffective for combining models.
- **Bayesian model combination**
  - Shows promise.

Monteith, et al. "Turning Bayesian model averaging into Bayesian model combination." Neural Networks (IJCNN), The 2011 International Joint Conference on. IEEE, 2011.